TABLE NO

STUDENT ID NO

MULTIMEDIA UNIVERSITY®

MULTIMEDIA UNIVERSITY

FINAL EXAMINATION

TRIMESTER 2, 2019/2020

TSN3151 – PARALLEL PROCESSING

(All sections / Groups)

10 MARCH 2020
9:00 a.m – 11:00 a.m.
(2 Hours)

Q1	
Q2	
Q3	
Q4	
Total	

INSTRUCTIONS TO STUDENT

1. This is an Open Book Exam. You are allowed to bring in any edition of *Parallel Programming* by Barry Wilkinson & Michael Allen, and/or 10 sides of A4 self-typed notes stapled together.
2. This Question paper consists of 10 pages with 4 Questions only.
2. Attempt **ALL** questions. The distribution of the marks for each question is given.
3. Please write all your answers in this question paper clearly.

Question 1 [25 marks]

a) Using the e-cube routing algorithm, what is the route taken in an 8-dimensional hypercube network from node 51 to node 200? Each step must be shown in node (decimal) number. Show all your steps clearly for full marks. [11 marks]

Continued...

b) Your employer has a local area network of 16 PCs, each with 16 processors. A large number-crunching job is needed to be calculated. The algorithm does not require neighbouring computations to communicate with each other, and the time needed for data transmission over the network is found to be 0.05% of the time needed for computation, on average. Should you implement this with MPI, OpenMP, or pthreads? Explain your choice and why the other two are not as suitable. [7 marks]

Which method:
Explanation:

c) Your employer has a PC with 1024 processors. A large number-crunching job is needed to be calculated. The algorithm does not need neighbouring computations to communicate with each other, but the time needed for data transmission over the network is found to be 50% of the time needed for computation, on average, because each computation is fast but is done on a large amount of data. Should you implement this with MPI, OpenMP, or pthreads? Explain why. [7 marks]

Which method:
Explanation:

Continued...

Question 2 [25 marks]

a) Change this serial program into an OpenMP program that would distribute the work in parallel properly. You can just indicate the line numbers you want to add or change, without having to recopy all the code. [9 marks]

```
 1 #include <stdio.h>
 2 #include <stdlib.h>
 3 #include <math.h>
 4
 5 int main(int argc, char *argv[])
 6 { double *data, result;
 7   int size, i;
 8   FILE *infile = fopen("2.dat","r");
 9   if (infile==NULL)
10   { perror ("Opening file");
11     exit(1);
12   }
13   fscanf(infile,"%d", &size);
14   printf("Size = %d\n",size);
15   data = (double *)malloc(sizeof(double)*size);
16   for (i=0; i<size; i++)
17     fscanf(infile,"%lf",&data[i]);
18   result = 0;
19   for (i=0; i<size; i++)
20     result += sin(data[i])+cos(data[(i+1)%size]);
21   printf("Result: %f\n",result);
22 }
```

Continued...

b) This code is supposed to take data from the user and then broadcast it to all processes so that each process knows what to process. The results are supposed to be bitwise-ANDed together. However, there are some mistakes.

```
 1 #include <stdio.h>
 2 #include <mpi.h>
 3
 4 int process(int, int);
 5
 6 int main(int argc, char *argv[])
 7 { int numprocs, rank;
 8   int data, result, finalResult;
 9
10   MPI_Init(&argc, &argv);
11   MPI_Comm_size(MPI_COMM_WORLD, &numprocs);
12   MPI_Comm_rank(MPI_COMM_WORLD, &rank);
13
14   if (rank)
15   { printf("Enter the integer data: ");
16     scanf("%d", &data);
17     MPI_Bcast(&data, 1, MPI_INT, 0, MPI_COMM_WORLD);
18   }
19   result = process(data, rank);
20   if (rank)
21   { MPI_Reduce(&result, &finalResult, 1, MPI_INT, MPI_BAND, 0,
MPI_COMM_WORLD);
22     printf("The final result of the calculation is
%d\n",finalResult);
23   }
24
25   MPI_Finalize();
26 }
```

What are the mistakes? [8 marks]

Continued...

Correct the mistakes (you can use the line numbers so that you do not have to recopy everything.) Assume that the function `process()` is provided somewhere. [8 marks]

Continued...

Question 3 [25 marks]

Change the serial program from question 2(a) into a pthread program with the number threads being the number of processes available that would distribute the work in parallel properly. You can just write things like "put code from line 1-4 here" instead of having to copy everything. [25 marks]

Continued...

Continued...

Question 4 [25 marks]

This serial program sums the sines of the numbers in the file.

```
 1 #include <stdio.h>
 2 #include <stdlib.h>
 3 #include <math.h>
 4
 5 int main(int argc, char *argv[])
 6 { double *data, result;
 7   int size, i;
 8   FILE *infile = fopen("4.dat","r");
 9   if (infile==NULL)
10   { perror ("Opening file");
11     exit(1);
12   }
13   fscanf(infile,"%d", &size);
14   printf("Size = %d\n",size);
15   data = (double *)malloc(sizeof(double)*size);
16   for (i=0; i<size; i++)
17   { fscanf(infile,"%lf",&data[i]);
18     printf("%f ",data[i]);
19   }
20   printf("\n");
21   result = 0;
22
23   for (i=0; i<size; i++)
24     result += sin(data[i]);
25   printf("Sum of sines: %f\n",result);
26 }
```

Change this serial program into an MPI program that would distribute the work in parallel properly. You can just write things like "put code from line 1-4 here" instead of having to copy everything. [25 marks]

Continued...

End of Paper